1-1-2005

An autonomous approach to proactive computer crime investigation

Noah Albert Korba
Iowa State University

An autonomous approach to proactive computer crime investigation

by

Noah Albert Korba

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Engineering

Program of Study Committee:
Doug Jacobson, Major Professor
Yu-Che Chen
Thomas Daniels

Iowa State University

Ames, Iowa

2005

Graduate College
Iowa State University

This is to certify that the master's thesis of

Noah Albert Korba

has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy

To my wife, Amy, and my son, Wes.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my major professor, Dr. Doug Jacobson, Sgt. Aaron Delashmutt, and all the investigators at the ISU Police for their support during the creation of Project Trident. Without their guidance and insight, Trident would not exist. I'd also like to thank my wife for her encouragement and support during my graduate career.

Chapter 1. Introduction

Crime committed over computer networks has increased dramatically over the past few years. From the illegal distribution of copyrighted material to Internet fraud, computer crime is becoming a prevalent entity in today's wired world. This type of crime has led to the advent of the computer crime investigation. This style of investigation shares many similarities with other types of investigations – a person has committed a crime, an investigator is assigned to figure out who (or whom) committed the crime, the investigator collects evidence and conducts interviews to determine who the suspect is, and, if enough evidence exists, the suspect is charged with the crime.

However, crime involving a computer has differences from a standard investigation as well – digital evidence can be easily faked or, by using commercially available products, removed completely. Interviews can be hard to conduct, as in many cases the involved parties exist outside a country's borders. Finally, proving presence can be nearly impossible, as computers are often made available to the public.

Solutions must be found to ensure digital evidence is discovered and collected before it has a chance to disappear. Such solutions should be proactive in nature, allowing authorities to discover the evidence on their own, rather than reactive, forcing authorities to respond to a crime involving a computer well after it is committed.

The goal of this thesis is to identify current methods being used to investigate computer crime, to offer criteria for the evaluation of autonomous approaches to investigating computer crime, and provide a solution for those that can benefit from a proactive, autonomous-based evidence collection tool.

Chapter 2. Background

In an extensive search for similar ideas to those presented in this thesis, only one proof-of-concept program has been developed. This program was developed is in conjunction with the Wyoming Internet Crimes Against Children (ICAC) Task force. The program, Peer Precision, is the centerpiece for a new attempt at stopping the exploitation of minors over the internet, called Operation Peer Precision.

Operation Peer Precision

Peer Precision was created and developed by Flint Waters of the Wyoming Internet Crimes Against Children task force[1]. Its purpose is to scour the Gnutella Peer-to-Peer network for known child pornography. It uses the open source Gnutella client PHEX as its interface to the network. Peer Precision works by first requiring the investigator to execute a query over the Gnutella client using PHEX, and then copy-and-paste the resulting SHA-1 hash values into Peer Precision. Peer Precision then compares these hashes to known child pornography hashes provided by several federal government agencies. If a match is found, Peer Precision then uses the IP address, in conjunction with the IP-to-geographical location database maintained at MaxMind.com, to determine where the source of the file is. Finally, to verify the content, the user must use PHEX to download the file.

The power of Peer Precision comes in its use of SHA-1 hashes collected from prior cases. A hash is like a unique fingerprint identifying a file. In previous cases involving child pornography, investigators have "hashed" identified images and movies of minors engaged in sexual acts. These hashes (or "hash sets") are made available to investigators across the country to aide in identifying illegal material quickly. Instead of paging through thousands of images on a computer containing evidence, an investigator can simply create hashes of all media on a computer, and then compare those hashes against their known hash set for illegal material. If any file matches, then the investigator knows for certain that the file contains illegal material. Peer Precision uses these hash sets against results returned when searches are initiated against the Gnutella network. A portion of the search results returned by Gnutella clients includes the SHA-1 hash of the file. Peer Precision compares the hashes returned by the search with the hash sets the investigator has to determine if any results contain illegal material. If so, the investigator can look further into what the computer is sharing on the network.

Operation Peer Precision has accomplished a lot in its trial runs. It has shown that authorities can use programs to actively look for illegal material being shared over peer-to-peer networks. It is also very accurate, as it uses SHA-1 hashes to verify content before it is downloaded, keeping authorities from wasting time looking at mislabeled files. However, Peer Precision does not maintain a database of shared information – the information provided is in real-time, so people sharing illegal material who are not on-line at the same time the officer using Peer Precision stay "under the radar".

Second, many officers are not well-versed in using peer-to-peer clients, leading them into potentially dangerous misuse situations. Officers may not completely understand where a download is coming from, causing them to create warrants based on inaccurate data. Finally, Peer Precision only covers one protocol, leaving others such as FTP and FastTrack wide open for the distribution of illegal material.

Though the Peer Precision program is a step in the right direction, it still fails to completely canvas computer crime taking place over networks. A better solution should be found to address these failures of Peer Precision, to ensure authorities are seeing the entire picture of computer networks, and are provided with accurate, reliable results to base warrants on.

Legalities: The Park Bench Scenario

Ideas like Peer Precision raise the question about the legalities of collecting public information provided by computers on file sharing networks. Such legalities can be summarized in the Park Bench Scenario. Consider a plainclothes police officer sitting next to two people having a conversation on a park bench. The two people, Paige and Andy, start talking about how they are going to sell drugs. Paige tells Andy that they can use her house to sells the drugs out of, and then provides Andy with the address of her house and the time they'll start selling. The police officer can then go back to the police station and draft a search warrant based on the conversation he overheard. This search warrant is completely valid, because the information in the conversation was provided in the public domain.

Computer networks, like the Park Bench Scenario, are also considered public domain. Though authorities cannot "tap" networks to listen into conversations without a wiretap warrant, as conversations between computers are considered private, they may see what information a computer is making public to the world. This information, usually in the form of files, can be accessed by anyone, including police. Ideas like Peer Precision take advantage of what computers make available, allowing law enforcement to draft search warrants based on what is being shared.

Chapter 3. Computer Crime Investigation

Computer crime has become a prevalent force in today's criminal investigations. Information such as child pornography and government ID templates have flooded file sharing networks. In fact, from July 1st, 2000 to June 30th, 2001 2,577 arrests were made for Internet sex crimes against minors[2]. To combat this, investigators have developed unique techniques in finding and prosecuting these criminals. However, before those methods are discussed, the typical method investigators use to investigate computer crime will be discussed.

Normally, a computer crime is investigated using a reactive approach – the crime is committed, the police are notified and respond, and an investigation starts. This approach can be classified as a "reactive investigation", or an investigation initiated by a citizen's notification. For instance, consider a theft from a store – a thief steals an item from a store (the crime), the store clerk calls the police (notification), the police come out and take a statement from the store clerk (response), and the officer passes the case to an investigator for further review.

However, computer crime is often committed without any sort of police notification. This can be partially attributed to the anonymity of the Internet – people can share information without authorities or other citizens ever seeing what is being shared. It is for this reason that, whenever possible, a proactive approach should be taken regarding computer crimes. A "proactive investigation" is where investigators attempt to catch a person committing a crime, without prior knowledge of who will actually commit the crime. For instance,

consider a sting operation involving narcotics – a police officer impersonates a drug dealer, a person approaches and asks to purchase drugs, the officer then sells the person drugs, and finally arrests the person for purchasing illegal drugs. This style of investigation is highly effective, but is very resource intensive, requiring time and money to complete.

The realm of computer crime is the perfect place to conduct proactive investigations. Only a small amount of resources are needed, these resources can be reused, and many times the investigation can be partially conducted autonomously – requiring no intervention by investigators.

Proactive Approaches to Computer Crime

Many different types of crime can involve a computer. A computer could be used to house drug sale records, to illegally trade copyrighted music, to coordinate the activities of terrorists, or to send threatening emails. However, many of these types of crimes, by nature, cannot be detected by proactive approaches. For instance, a threatening email becomes a crime only when the victim reads it and determines it a threat. It is for this reason that only specific computer crimes lend themselves to proactive investigations.

One of these types of crimes – the distribution of child pornography – is an especially heinous crime. It involves the transfer of images and videos of minors engaged in sexual acts. It is said that, every time an image is transferred, the minor is victimized again. With

thousands of these transfers taking place every day, this victimization is widespread, and hard to control.

The US government, however, has taken steps to combat this crime. The National Center for Missing and Exploited Children (NCMEC) has set up a tip line for users to call when child pornography is found. The FBI, in coordination with other local agencies, set up stings involving computers hosting child pornography. Once the child pornography has been downloaded, the investigators trace back those who received the illegal material. These proactive solutions, in conjunction with standard investigation techniques, led to over 2000 prosecutions between 2000 and 2002[3], not including arrests.

However, stings take large amounts of time and resources to execute, and require deep coordination between federal, state and local agencies. The NCMEC tip line also requires time and resources, and investigators have no control over the quality of information provided – a citizen could abuse the line, taking away precious time from investigators to chase non-existent problems. Clearly, this "man-power" approach to proactive computer investigation works, but at a great cost.

Therefore, a tool should be created that can conduct an autonomous sweep of computers, to find those sharing illegal material. Such a tool should alleviate resource constraints of sting-based operations, by continually scanning for and identifying known illegal material. The tool should be controlled by the investigators, to ensure the quality of evidence, and, unlike tip lines, ensures that the investigators are spending time on known criminal violations.

Using the Best Practices Guide for Seizing Electronic Evidence[4], four criteria were developed to evaluate such a tool. These evaluation criteria are modeled after steps taken during a standard search and seizure of physical electronic evidence.

Independent and Legal Operation

To reduce both the intellectual and resource constraints of investigators, the tool should not require any intervention from law enforcement to operate. This will make it easier for investigators to access the information they need, without having to spend precious time learning about the inner-workings of network protocols and server maintenance.

In addition, the tool should stay within legal boundaries when accessing information. This ensures the information provided by the tool can be admissible into court proceedings, as well as any warrants.

Preservation of Evidence Integrity and Accuracy

The tool should maintain computer forensic integrity while scanning and reporting. This is essential in computer investigations due to the fragility of computer evidence. If the tool modifies the questionable material in any way, it may not be admissible in any type of prosecution.

The tool should also maintain the integrity of the information being presented to the investigator. Malformed or inaccurate information can invalidate search credentials (such as a search warrant), which may result in a case being discarded.

Information Accessibility and Quality

The tool should provide enough information for investigators to correctly identify the illegal material and obtain proper search credentials to lead to an arrest. The method used to present the information should provide for minimal understanding of the underlying network protocols. This will allow investigators to view the information, without requiring them to have extensive information on protocol architectures.

In addition, the tool should ensure that the quality of the information is not degraded during transmission. The tool should present the information at the same quality as it exists on the suspect computer, without interference. This ensures the investigator will be able to determine, on their own, the legality of the information being presented.

Finally, the tool should not be limited within the current state of the network. The tool should keep track of both current and past states, as computers join and leave file sharing networks quite often. This ensures authorities are given the complete picture of the network – no matter what time it is.

Protocol Adaptation

The tool should be able to provide access to information over multiple network protocols. As new network protocols come to light, the tool should allow investigators to search over these new protocols with the same ease as older, established protocols. This gives the tool flexibility and longevity, preventing it from becoming obsolete.

The program "Peer Precision" only meets two of these criteria. Peer Precision ensures accuracy with the results returned via hash value comparison, and it allows authorities to download and verify suspected material. However, Peer Precision requires a constant user presence and interaction to copy-and-paste hash values in, forcing authorities to learn how to use two new interfaces. Second, Peer Precision only works over the Gnutella network, completely disregarding any other type of file sharing network. This severely limits the scope in which authorities can investigate on-line criminal activity. Finally, Peer Precision only provides the current state of the Gnutella network. Any users in the past have "missed" the watchful eye of law enforcement. This keeps law enforcement from seeing the entire picture of the Gnutella network.

Therefore, a tool should be created to meet all four of these criteria. The software should be easy to use, accurate, flexible, and require no user interaction for operation. By implementing these criteria, the tool will have a better chance at being adapted by authorities for daily use in investigations.

Chapter 4. Project Trident

Project Trident is one tool created to accomplish proactive autonomous computer crime investigations. Trident is designed to be complex enough to scan different types of network protocols for available information, but easy enough for any investigator to use. Trident was created in a modular fashion, using input modules to conduct individual protocol searches. This allows Trident to adapt to new technologies quickly – a new module is created for every new protocol, rather than having to rewrite the entire interface.

Trident is broken into three parts – the indexing engine, the search engine and the browsing service (Figure 1). The indexing engine scans different file distribution networks, and indexes the files available on these networks. The search engine provides a simplified interface to the results generated by the indexer, and allows a user to execute a keyword search against the results. The browsing service provides a seamless web interface to any file distribution network Trident indexes, allowing a user to browse and download files from a host.

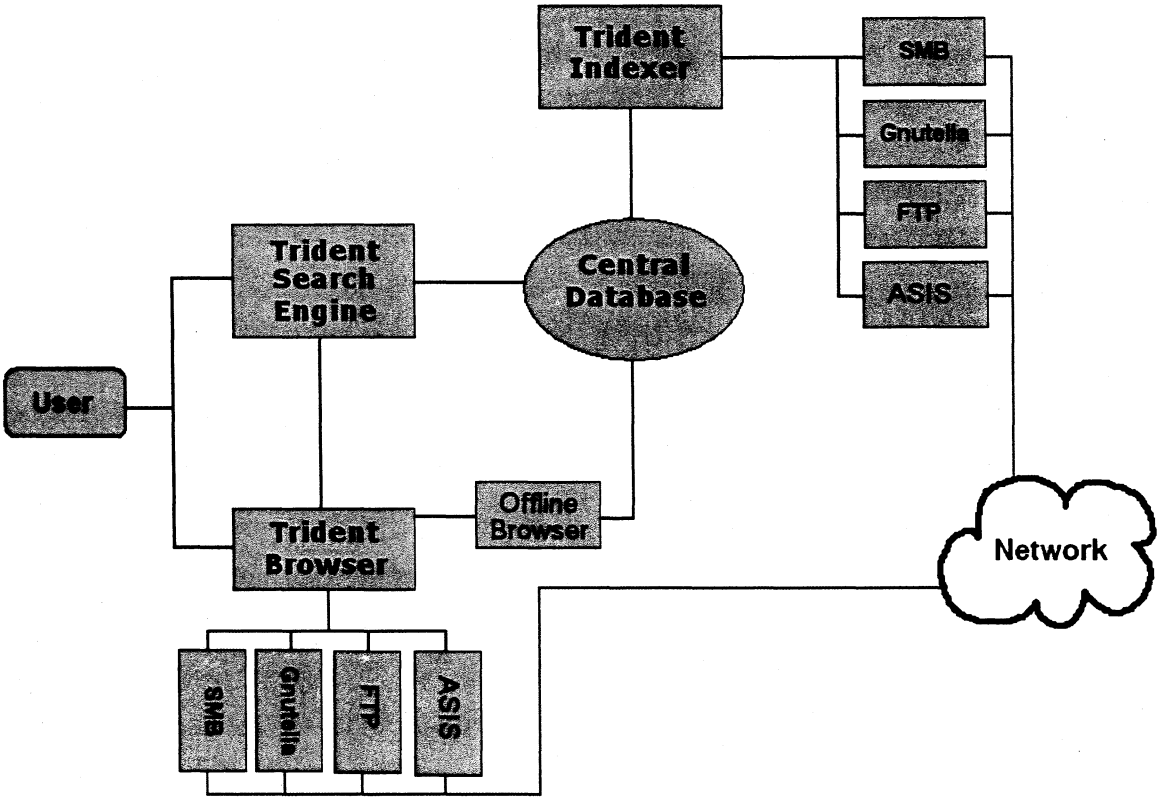

Figure 1 - The overall design of Project Trident

According to Nielsen//NetRatings, approximately 59 million people worldwide[5] use the Internet. As such, Trident's front end was created as a web-base client in order to appear to this large user base. This allows anyone with a web browser to search through available network resources and view those resources when appropriate. Trident is designed such that no network protocol knowledge is needed – only the ability to search for and discover illegal material over a given network. If the user can search the Internet, they can use Trident.

Trident satisfies all four characteristics previously defined. The Trident indexer runs autonomously, keeping track of the past and current state of the network, and the files being

shared within. Trident operates read-only over the network; ensuring files are not changed by the indexer (thus preserving the integrity of the file). The indexer and browser access information using "guest", "public" or "anonymous" accounts, ensuring that the information provided could be seen by anyone in the world. The Trident search engine and browser are all web-based, allowing anyone with Internet browsing experience to search and view files located by the indexer. Finally, the indexing and browsing components are modular based, allowing Trident to easily adapt to new network protocols.

Chapter 5. The Trident Network Indexer

Trident was designed to work with many different types of file distribution networks. As such, Trident is designed to be modular – composing of the main indexing engine that manages what IP addresses to scan, and individual service modules. When started, the Indexer loads a set of IP addresses from a configuration file created by the user. It then passes each IP address to every module. Each module scans a computer for files being offered by a particular service. If any files are found, it will report back to the main indexing engine, in a common format, what files are available on that computer. The main indexing engine will then enter this information into a database table that the search engine can query.

Modules

Each module can be passed only one argument – the IP address it is supposed to scan. When invoked, the module first passes the primer line, which passes configuration and identification information to the indexer (Table 1). This line is also used by the search engine, and is referenced when the user wishes to browse for a file or directory.

Table 1 - Primer line

Field	**Purpose**
Database	The name of the database the information should be stored in
Short Name	The short name of the module
Long Name	A more descriptive name of the module
Browser	The browsing service to be used when the user wishes to connect to the computer

Once the module has been passed the IP address, it scans the computer for files being shared on its network. It then passes back to the indexer information about each file it encountered on its sweep. In order for Trident to be extensible, a common reporting format was needed to simplify and quickly organize the information for the search engine to process (

Table 2). This reporting technique provides each module the ability to provide both standard information, such as paths and sizes, and non-standard information, such as ports and login information, back to the indexer. Non-standard information is stored in the "Extra" field, and is ignored by both the search engine and the indexer. This extra information is passed to the browser without any modification, allowing the module to communicate with the browser without any interference from Trident.

Table 2 - Information fields reported by each module

Field	Purpose
IP	The source of the file
Host	The host name of the IP on the file distribution network
UID	Any user identity provided by the file distribution network
Path	The path to the file
File	The file name
Size	The size of the file
Extra	Any extra information that should be passed to the browser when referencing this file (such as port numbers or login information)

Post-Module

After Trident invokes the module, it captures the file listing provided by the module and stores it into a table named after the IP inside the database provided in the module's primer line. Once completed, Trident performs any optimization needed on the table within the database. Next, it updates a master table index entry (Table 3), which contains statistical information about the files being provided by every IP over the file distribution network it scanned (this information is referenced by the search engine). Finally, Trident scans the IP

with the next module in the list. Once all modules have been executed, Trident moves to the next IP and the process starts over.

Table 3 - Master Table Index (MTI) format

Field	**Purpose**
IP	What IP address the information relates to
Host	The host name of the IP on the file distribution network
UID	The user identity provided by the file distribution network
Last Date	The last time this IP was scanned (used when phasing the database)
File Count	How many files this IP has made available over the file distribution network

Indexer Modules

SMB Module

The SMB module was the first module developed for Trident. It allows Trident to communicate with Windows sharing (NetBIOS) networks. It uses a modified version of the SMB client provided with Slackware Linux. This modification (called "xsmbclient") changes the way the standard SMB client reports back information. Rather than having a human-

readable format as output, the xsmbclient reports back in a colon-separated form. This allows the SMB module to wrap the output of the SMB client, making it easier to parse out the information being passed back. A colon was chosen as the delimiter, as file and paths are not allowed to have the colon character as a part of their names, making the colon a uniquely identifiable character.

The module also uses the "nbtscan" program to quickly retrieve SMB-related identifying information (Table 4). This information is reported with every file reported back to Trident, ensuring it is stored for later recovery. The host identifier is also used when connecting to the computer, as it is a requirement to know the NetBIOS host name when accessing shares on an SMB-based network.

Table 4 - Information retrieved via "nbtscan'

Identifier	**Explanation**
Host	This is the NetBIOS host name associated with the IP address. Users will often label this using a personal identifier, such as their last name or the type of computer it is.
UID	This is the ID of the current user logged in. Many times users will use an ID with either their first or last name, or a combination, such as "jsmith" or "johns". This is equivalent to the Windows login name.

When initiated, the SMB module connects using a guest username and no password over port 139 to the IP address given by Trident. Using only a guest user name and no password ensures it only gathers publicly available information. Once it connects, it executes a directory listing command on the remote host via the xsmbclient. It logs any files and their sizes, and proceeds to scan subdirectories recursively until every directory on the remote host has been indexed. This information (the path, file name, and size) then gets passed back to Trident for final indexing, after which the module exits.

The SMB module was written the Practical Extraction and Reporting Language (PERL). PERL was chosen as it is a very portable language, is quick to develop in, and is made for program wrapping and data parsing. The xsmbclient was written in C, as the original SMB client suite was written in C, and the xsmbclient is just a modified version of that source. The SMB client and the xsmbclient are both under the Gnu Public License (GPL). The nbtscan program is also released under the GPL. The Iowa State University Office of Intellectual Property and Technology Transfer has copyrighted the SMB module, and retains licensing rights to the software.

The Gnutella Module

The Gnutella module provides an interface to the Gnutella peer-to-peer network. It connects natively to Gnutella hosts, not requiring any outside software. The Gnutella agent only indexes computers that allow other computers the ability to retrieve a "host listing" from

what they have shared. As such, the module does not have to login to any peer-to-peer network to view the contents of a computer. This allows Trident to have a one-to-one relationship to the peer-to-peer node.

The Gnutella module does not retrieve any uniquely identifiable host information, as the Gnutella protocol does not provide any information that could be used to uniquely identify a user. A user should use the files made available via the Gnutella peer to identify it, as well as any records relating to the IP address of the peer. The Gnutella module does not report a user ID, and only reports the DNS name relating to the IP address.

When invoked, the Gnutella module connects over port 6346 (the standard listening port for Gnutella) to the IP address given by Trident. It then issues the following request:

```
GET / HTTP/1.0
User-Agent: Gnutella
Host: 129.186.215.173:6346
Connection: Keep-Alive
Accept: application/x-gnutella-packets
```

This requests a Gnutella-encoded version of what the peer is sharing on the Gnutella network. If the peer allows listings, it will return the listing and close the connection. The module will then parse out the file information from the encoded packet, and report the listing to Trident. Since Gnutella does not use any directory structure for searching, no

recursion is necessary to parse the peer's share records. The file name, file size and file index are provided to Trident for indexing.

Though the Gnutella protocol does not use a directory structure, directories are still reported back to Trident. This is due to how file requests are made via the Gnutella protocol. When requesting a file, the file's index and name are provided to the Gnutella peer. When parsing the reply from the Gnutella peer, the module also extracts the file's index number. This index is stored in Trident as the file's path. This allows the Gnutella browser to correctly request the file.

The Gnutella module was written in the Practical Extraction and Reporting Language (PERL). This was chosen due to PERL's phenomenal ability to quickly parse different types of information. The Iowa State University Office of Intellectual Property and Technology Transfer has copyrighted the Gnutella module, and retains licensing rights to the software.

The FTP Module

The FTP module provides Trident with an interface to File Transfer Protocol (FTP) servers. The FTP module uses NCFTPLS, a freely available command line interface to FTP servers. NCFTPLS was used to aide in the rapid development of an FTP module for Trident, and, at some point, will be replaced with a native interface to the File Transfer protocol. NCFTPLS

allows both standard (client-burdened) and passive (server-burdened) connections to FTP servers, allowing it to communicate with various FTP server configurations.

Like the Gnutella module, the FTP module does not gather any uniquely identifiable information. The FTP server does not present any user ID when connecting. The user should use the contents of the files to determine the server administrator's identification. The host name reported by the module is simply the DNS name of the IP address.

When initiated, the FTP module will connect to the IP address provided by Trident over port 21. It will attempt to login anonymously (user name of "anonymous" and password of "wget@"), which ensures the module is only accessing publicly available information. It then requests a directory listing, where directory names, file names and file sizes are extracted. The file names and sizes are reported back to Trident. If the listing provides a directory, the module will then scan that subdirectory. This process will continue until every subdirectory has been scanned. The module will then report the results to Trident, and exit.

The FTP module was written in the Practical Extraction and Reporting Language (PERL). This was chosen due to NCFTPLS's reporting style, and PERL's ability to quickly parse NCFTPLS's directory listing. The NCFTPLS program was originally written in C. It is released under the Gnu Public License (GPL). The Iowa State University Office of Intellectual Property and Technology Transfer has copyrighted the FTP module, and retains licensing rights to the software.

The ASIS Module

The Advanced Sharing Indexing Service (ASIS) module allows Trident to retrieve information about computers on peer-to-peer networks that do not allow direct querying of information (such as the Fast Track network). The ASIS module interfaces with the ASIS daemon.

The ASIS daemon was created to simulate a client-server connection to peer-to-peer networks that do not support direct connections, such as the Fast Track network. The ASIS daemon has a list of known peer-to-peer networks, and sequentially connects to each one. It then loads a list of known child pornography terms (provided by the FBI), and executes a search on the network for each one. Next, ASIS takes the results (Table 5) from this query, and stores them into a database. Once ASIS reaches the end of the network list, it starts back at the top, and the process continues.

Table 5 - Results from an ASIS query

Field	**Purpose**
IP Address	This is the IP address of where the result originates.
Host	This is the host name associated with the peer-to-peer network. This is usually the DNS name of the IP address.
UID	This is the user ID associated with the result. This is usually the name of the client the peer is using to access the network.
Path	The path to the file
File Name	The name of the file
Size *	The size of the file
Date *	The date of the file
Hash Type *	The type of hash used to uniquely identify the result
Hash Value *	The value of the hash associated with the file
Last Date	The time this result was entered

* Denotes entries that are optional

When invoked, the ASIS module connects to the ASIS daemon, and queries for all records pertaining to the IP address Trident passes to the module. This allows Trident to simulate a worldwide peer-to-peer connection. Instead of having to connect to every peer-to-peer network across the globe, and search for keywords to generate results, the module simply queries a large record service that is continually crawling these networks. This allows the module to work very efficiently. It also simulates a client-server network, which is

conceptually easier to understand for regular users. In essence, the module is connecting to the "ASIS file-sharing network".

Once the query has completed, the ASIS agent relays the host, user ID, path, name and size of every file located by the daemon relating to the IP address passed by Trident. Trident inserts this information into its index, and the module exits.

The ASIS module was written in the Practical Extraction and Reporting Language (PERL). This was chosen due to PERL's ease in portability. The ASIS daemon is a combination of the giFT peer-to-peer server, a modified version of the giFT command-line interface, and a centralized scripting, querying and database management program.

The giFT server is released under the Gnu Public License (GPL). The modified giFT command-line interface is also released under the GPL. The ASIS daemon is not released under any license. The Iowa State University Office of Intellectual Property and Technology Transfer has copyrighted the ASIS module, and retains licensing rights to the software.

Chapter 6. The Trident Search Engine

Trident's search engine provides an easy interface to the millions of file records stored by Trident's indexer. The search interface allows users to restrict the results to a particular module, only display image or video files. It also lets the user to show results only from a particular IP address.

When a basic query is launched, Trident executes the keyword search across both the file paths and file names stored in each database's table. The results are organized first by module, and then by IP. A sample result is showed for each IP that has files that match the search query (Figure 2), along with information pertaining to the host that possesses the file.

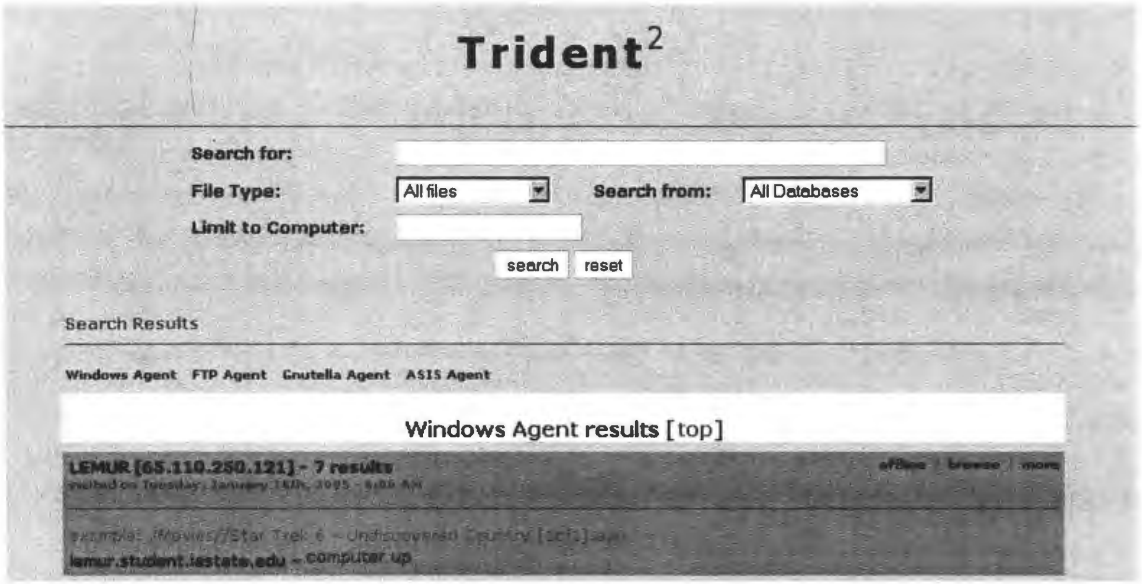

Figure 2 - Example search result

The search engine also checks to see if the computer is available by "pinging" the computer. If a response comes back, the computer is considered to be "up". Otherwise, the computer is said to be "down". This method is not entirely accurate, as a computer may be up, but it may not be currently connected to the file distribution network the result is coming from.

Once a result of interest is found, the user can either click on "more", which will show more results from that host, or "browse", which allows the user to browse the particular network the results are coming from. If available, a user may also click on "offline", which invokes the off-line browser.

Chapter 7. The Trident Generic Browsing Interface

To simplify the browsing process, a generic interface was developed, keeping the user from having to understand the inner workings of different protocols. The goal of the Generic Browsing Interface (GBI) was to create an interface that conforms to how users think about computer files. This interface - consisting of folders and files users can click on - allows module creators to focus solely on interfacing to file distribution networks, rather than having to focus on creating a friendly user interface. The interface also handles advanced options, such as thumbnail viewing, and file type filtering (see Figure 3).

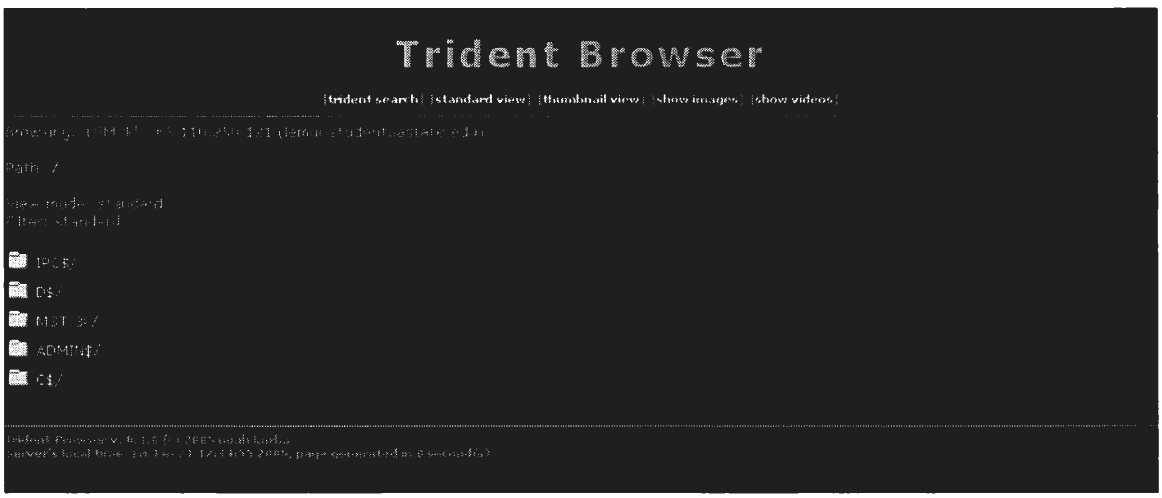

Figure 3 - Generic browsing interface

Browser modules, or "viewers", work similarly to the indexing modules. A standardized format was used to handle viewer input and output. The browsing destination (target) is

provided, along with other values, to the viewer (Table 6). The viewer, in turn, provides the requested output.

Table 6 - Viewer input parameters

Field	Purpose
IP	The IP address of the target
PATH	The path to browse
FILE	If specified, the file to download
EXTRA	Any information stored with the PATH from the corresponding indexing module
CHECK	If specified, the viewer is to check the status of the file.

If the FILE argument is not specified, the viewer will produce a directory listing for PATH. The output from the viewer is rendered verbatim to the user, so it is up to the viewer to produce the results in a sorted fashion. The viewer presents the list in rows, where each row corresponds to either a folder or a file. Each row is a colon-separated list, ordered as described in Table 7. The GBI will render this listing to the user, and will make each file or folder "clickable", so the user may navigate to a subfolder, or download a file within the path.

If the FILE argument is passed to the viewer, but the CHECK argument is not specified, the viewer simply starts streaming the file to standard out. When the user first clicks on the file, the GBI requests a check from the viewer, to ensure the file is available for download. If it is available, the GBI then analyzes the file's extension, and attempts to prepare the user's browser for the specific file type. For instance, if the user requests the file "some_image.jpg", the GBI identifies the extension "JPG" as a JPEG image, and sends the HTTP content header, "image/jpeg", along with the file size and name, to the user's browser. If no type match can be made, the GBI simply identifies the file as an "octet-stream", and proceeds with the download.

Once the headers have been created and sent, the GBI initiates the file download with the viewer. As described before, the viewer prints the file information to standard out. The GBI collects this information, and starts streaming it to the user. To the user, this entire process looks simply like a download from a website – they are never involved in the inner-workings of the specific network protocol they're browsing.

If the FILE and CHECK arguments are specified, the viewer will check the status of a file. If it's available for download, the viewer returns the value 1. If not, the viewer returns 0. This is primarily used before the download begins.

Table 7 - Viewer output parameters

Value Position	Purpose
0	Username
1	Protocol-specific hostname (often the DNS name of the IP address)
2	Path to value 3
3	File or folder name
4	The size, in bytes (if it's a file)
5	The date, in seconds from the Epoch (UNIX time)
6	Type – a "D" for a folder, or an "F" for a file

Browsing Modes

The GBI supports two types of rendering, and two types of filters. Standard, the first type of rendering (as shown in Figure 3) – shows files and folders similar to that of a file system listing, where files have a "file" icon, and folders have a "folder" icon. The second type of rendering – thumbnail – shows files and folders in standard mode, unless they're images, in which case they are rendered as thumbnails (Figure 4). This mode allows users to quickly browse suspected images, without having to repeatedly click, view and go back to the previous web page.

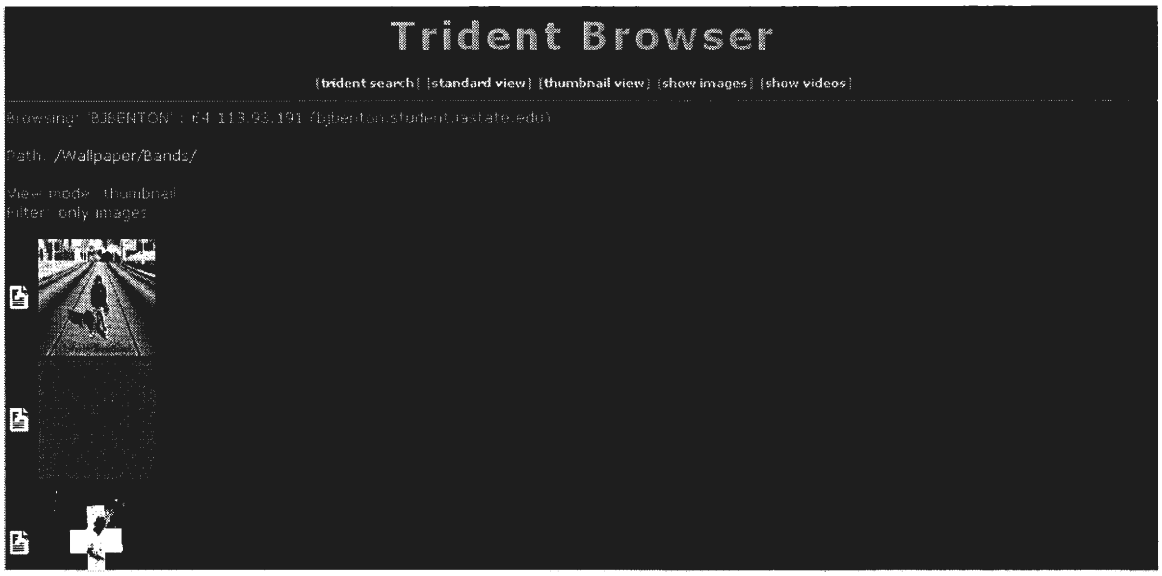

Figure 4 - Thumbnail rendering mode

The GBI also supports two filtering modes. The first filter – images – will show only images in the folder. The second – video – will show only videos in the folder. These filters allow users to filter out unwanted material, which is important if the folder they're browsing contains many unrelated files.

Offline Browsing

The off-line browser is an extension of the Trident indexer. It allows a user to browse for files as if the machine was on-line. It works by connecting to the indexer's database, and recreating a computer's file system as it were when Trident last scanned the computer. This helps users see the entire picture, without having to wait for the suspect computer to come on-line.

However, the off-line browser does not paint a complete picture. If the information being shared by a computer changes after the indexer visits, the user could be seeing an out-of-date file structure. Also, the user can only see file and folder names – the actual content is not cached by the indexer, and subsequently, not available for off-line viewing. The user will still have to wait to view the content until the computer comes on-line.

The off-line browser can also render file systems in a confusing nature. For instance, when the off-line browser attempts to render a computer's file system retrieved over the Gnutella network, the user is presented with a list of folders, each containing one file (Figure 5). This is due to how the indexer stores the results from the Gnutella module. Since Gnutella uses file indexes along with file names for retrieval, the off-line browser thinks this indexes as folders, and renders them as such.

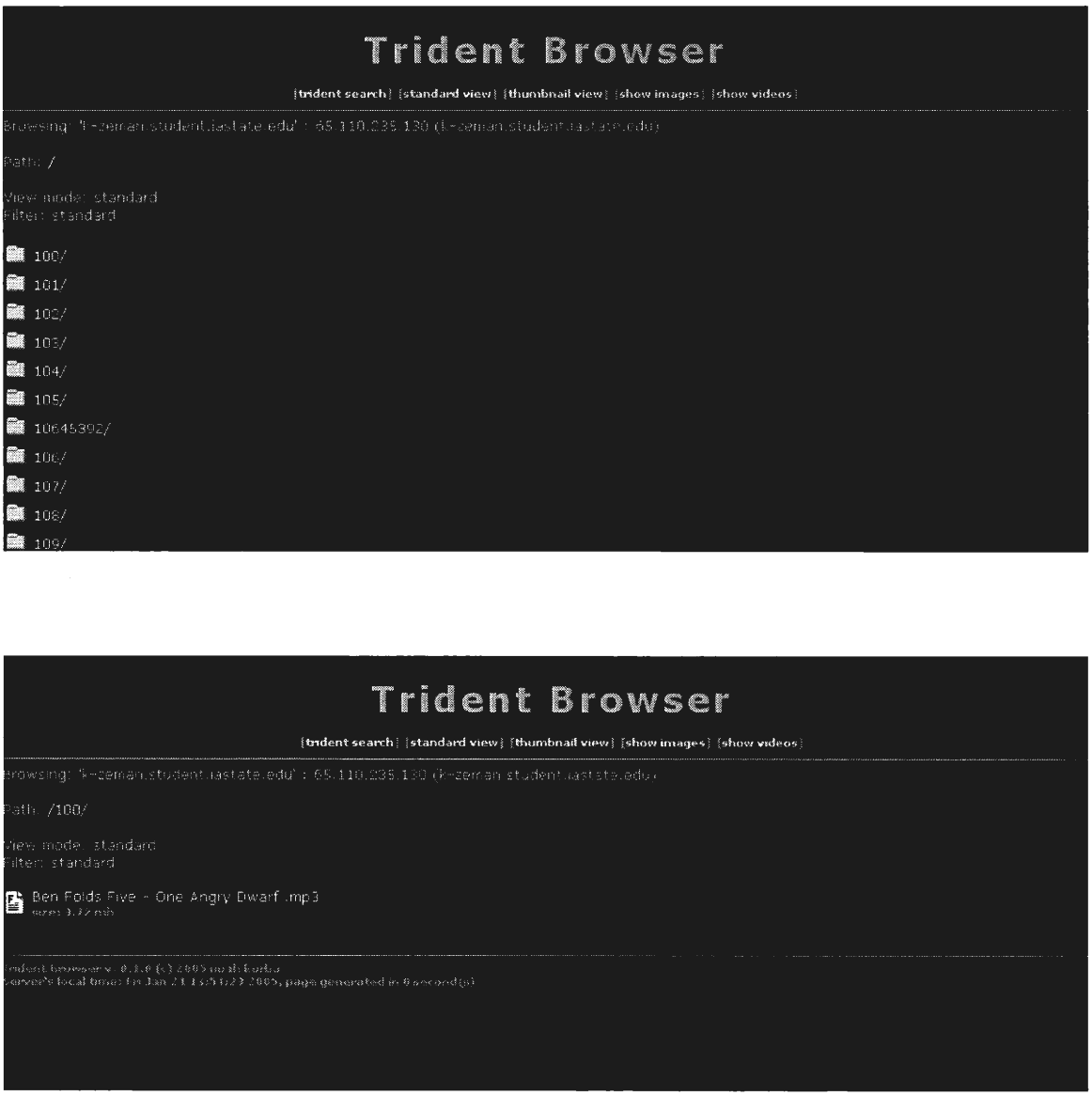

Figure 5 - Confusion caused by off-line viewing

Though the off-line browser can render information in a confusing manner, it still remains an important tool for users to have. Off-line browsing can help a user discover related files, and prepare them for when the computer appears back on-line.

Chapter 8. Testing and Verification

To test Trident out, three computers were set up on a network. Unique files were placed on these computers (named "trident_file"), and each computer was set up to offer one of three protocols – Gnutella, SMB or FTP. Trident was then instructed to scan these three computers. Next, the search interface was used to find these three files by using the keyword "trident". Finally, the Trident browser was used to download and verify the contents of these files.

The Gnutella Protocol

The first protocol tested was Gnutella, using Bearshare to interface to the protocol. The file "trident_file.txt" was placed in Bearshare's "shared" folder on the computer "PTEST", which has the following contents:

```
Test Trident File for Bearshare
```

Trident then scanned the computer over all protocols, and stored the share information of the computer into the database. Next, the Trident search engine was used to launch a query for "trident". The result (Figure 6) returned showed the computer "PTEST" matching with the file "trident_file.txt".

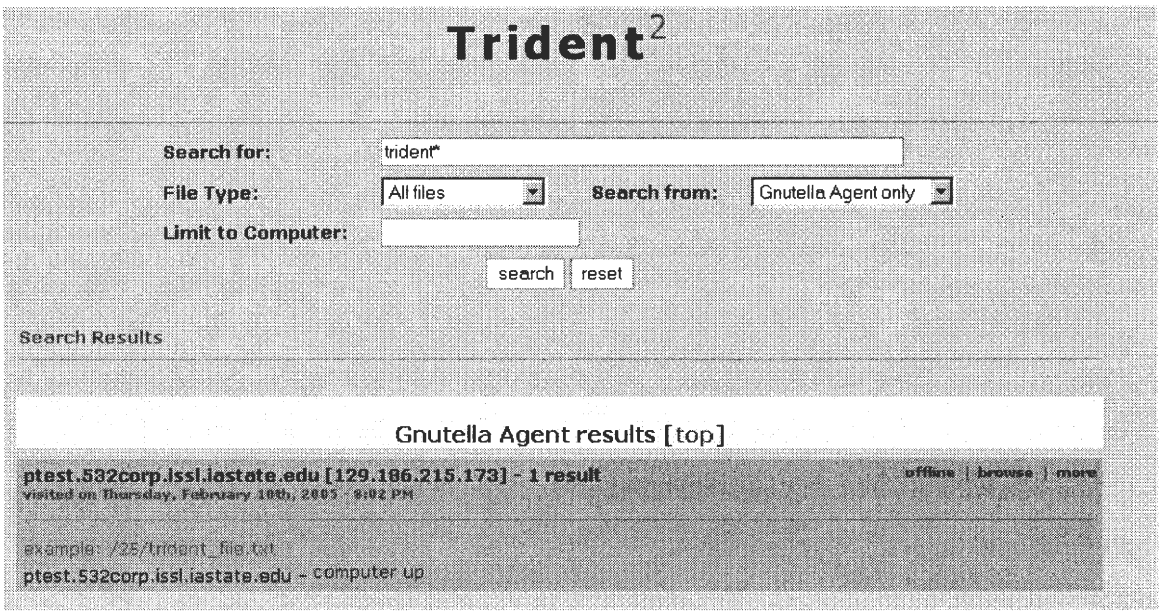

Figure 6 - Query results from the Gnutella module

Figure 7 - Browsing the Gnutella test computer

The Trident browser was then used to connect to the Gnutella computer, and retrieve a file listing (Figure 7). Finally, the file was located and downloaded for view. As expected, the file contents match exactly to what was placed in Bearshare's "shared" folder (Figure 8).

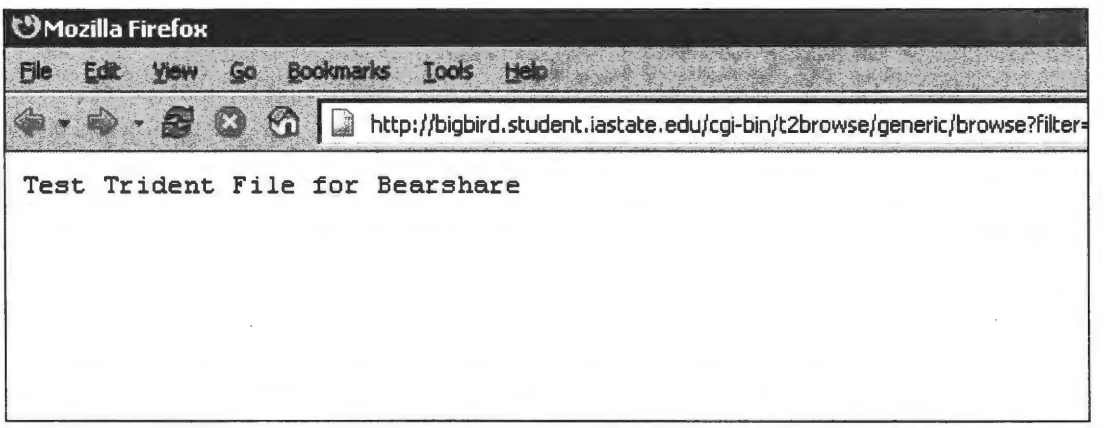

Figure 8 - The file placed on the Gnutella server

The SMB Protocol

The second protocol tested was SMB, using UNIX's Samba server to interface to the protocol. The file "trident_file.txt" was placed in a publicly accessible folder on the server "BIGBIRD", which has the following contents:

```
Test Trident File for SMB
```

Trident then scanned the computer over all protocols, and stored the share information of the computer into the database. Next, the Trident search engine was used to launch a query for "trident". The result (Figure 9) returned showed the computer "BIGBIRD" matching with the file "trident_file.txt".

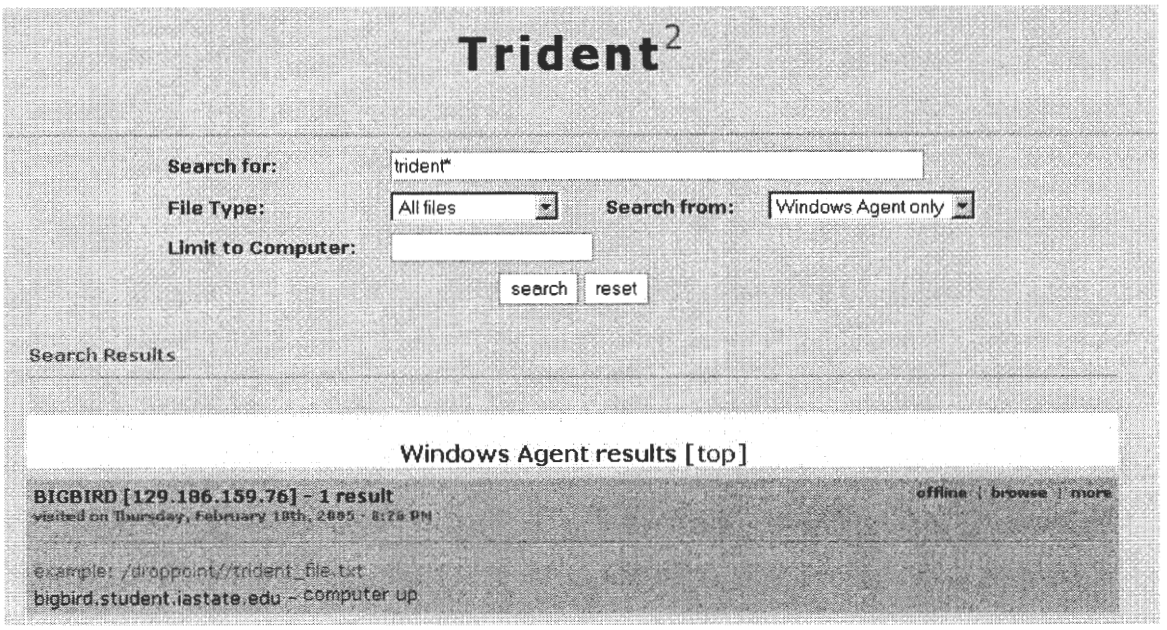

Figure 9 - Query results from the SMB module

The Trident browser was then used to connect to the SMB computer, and retrieve a file listing (Figure 10). Finally, the file was located and downloaded for view. As expected, the file contents match exactly to what was placed in BIGBIRD's "shared" folder.

Figure 10 - Browsing the SMB test computer

The FTP Protocol

The last protocol tested was FTP, using UNIX's FTP server to interface to the protocol. The file "trident_file.txt" was placed in a publicly accessible folder on the server "WORF", which has the following contents:

```
Test Trident File for FTP
```

Trident then scanned the computer over all protocols, and stored the share information of the computer into the database. Next, the Trident search engine was used to launch a query for "trident". The result (Figure 11) returned showed the computer "192.168.1.53" (WORF's IP address) matching with the file "trident_file.txt".

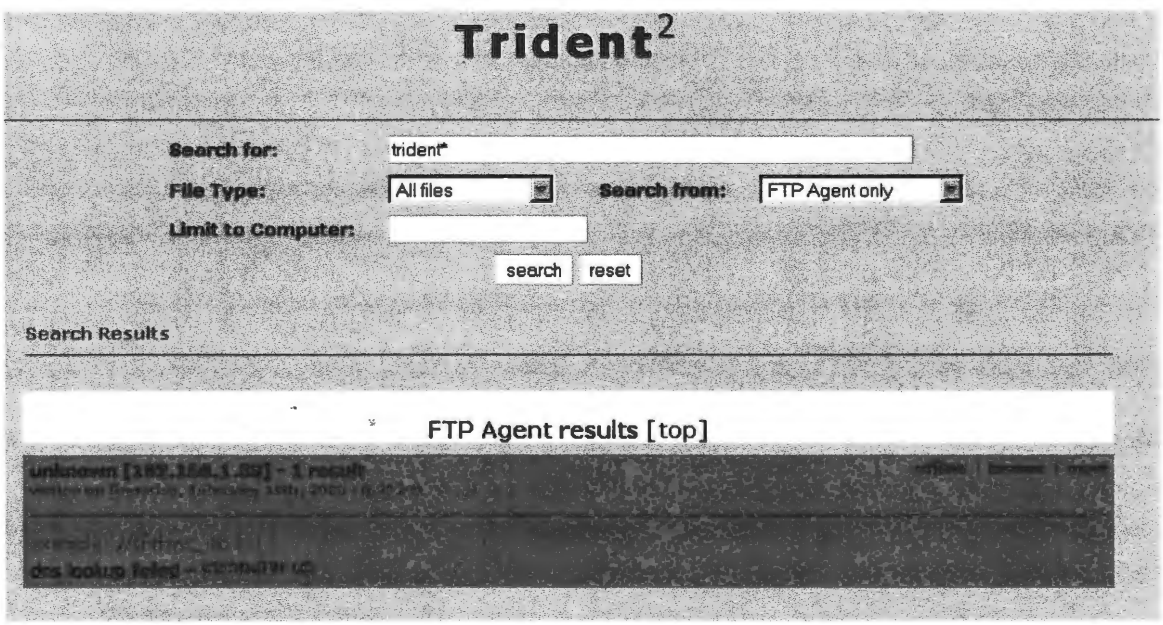

Figure 11 - Query results for the FTP module

The Trident browser was then used to connect to the SMB computer, and retrieve a file listing (Figure 12). Finally, the file was located and downloaded for view. As expected, the file contents match exactly to what was placed in WORF's "shared" folder.

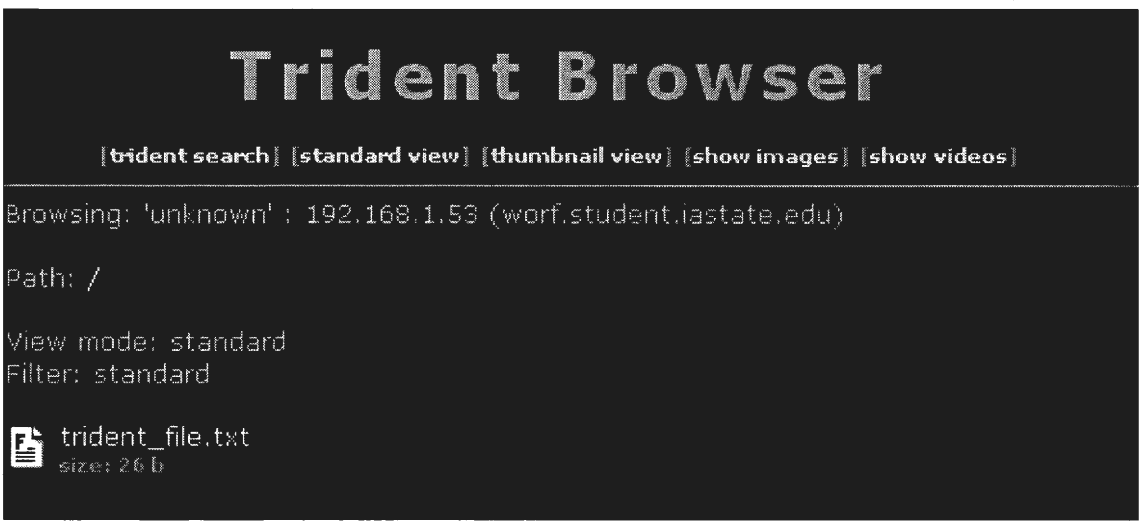

Figure 12 - Browsing the FTP test computer

Chapter 9. Extended Applications of Trident

In addition to Trident's originally intended application, two other applications for use have been identified – MAC address caching and user name extraction. Both applications were derived from the information provided by the SMB module. They will be discussed below.

MAC Address Caching

When the SMB module queries a computer for its basic information, it retrieves, among other things, the computer's MAC address. As a part of the functionality of the SMB module, this MAC address is cached with the other basic information about the computer into the Master Table Index. When computers are stolen, thieves may start using the computer on the network by changing the computer's name and/or IP address. This process makes it very difficult to digitally track the activities of a stolen computer. However, in the Microsoft Windows™ environment, the SMB implementation provides the MAC address to the SMB module. This means that, even if the thief changes the name or IP address on the network, Windows™ will return the MAC address of the network card, giving a hardware-level piece of information for investigators to search by.

This discovery led to the SMB/MAC search interface, which is a search engine available within Trident to search for computers matching a MAC address. This interface, provided through a web page, allows users to type in all or part of a MAC address, and Trident will

display computers that match this address. This allows investigators to quickly determine if a stolen computer is still present and being used.

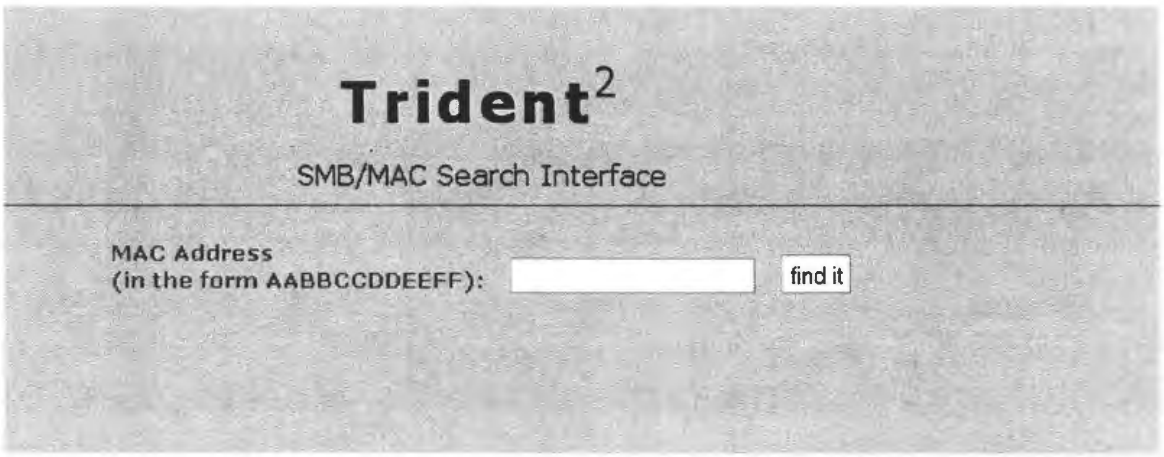

Figure 13 - SMB/MAC search interface

To test, a computer was placed and configured to run on a sample network (Table 8). Trident was then instructed to scan this IP address, using all modules available. Once completed, the SMB/MAC interface was used to locate the computer by its MAC address. Figure 14 shows that the SMB/MAC Search interface was able to correctly match the computer's MAC address with the computer's IP address. From here, investigators can use Trident to see what the computer is sharing, or start making plans on how to recover the computer.

Table 8 - SMB/MAC test environment

Computer Name	Computer IP	MAC Address
Mrsparrow	129.186.159.243	00:0B:DB:A0:3A:B5

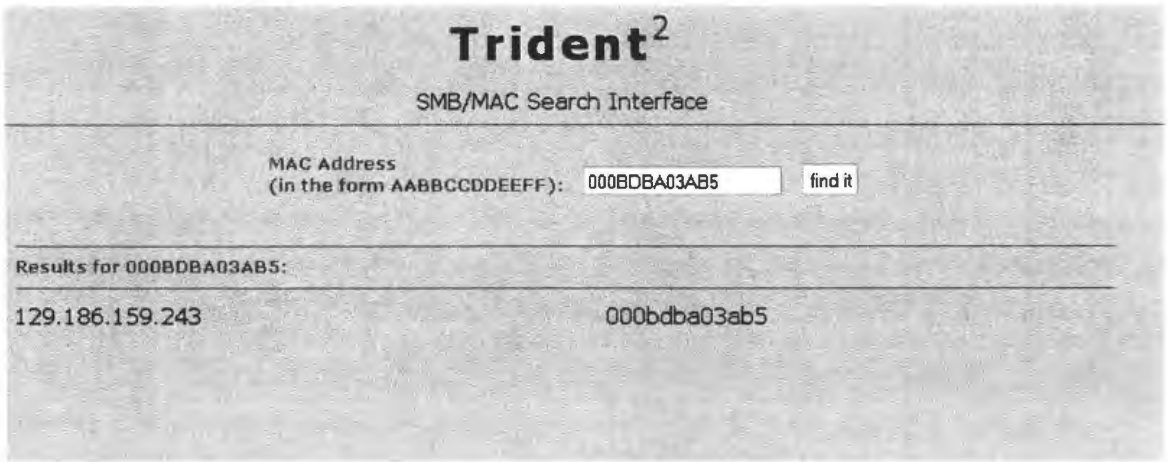

Figure 14 - SMB/MAC search results

The SMB/MAC search interface has two drawbacks. First, it only works with computers running Microsoft Windows™. UNIX implementations of the SMB protocol (such as Samba™) return all 0's as the MAC address. This helps investigators identify the type of operating system present on the computer, but fails to help them in computer thefts. Also, if the thief spoofs the MAC address (which can be done on some network cards), this search would be rendered useless.

User Name Extraction

The SMB protocol also provides user information when initially scanned. This identifies the user currently logged into the computer. Since the SMB module was already storing this information inside the Master Table Index, a simple search application was created to facilitate quick, web accessible queries to the SMB module's MTI. This allows investigators

to search for specific user identifiers across the network, hopefully being able to tie a computer to a person.

This user identification is not an exact science. In fact, in some situations, the SMB module is not able to extract any user name at all (as shown in Figure 15). Users can also provide "fake" information, causing Trident to record inaccurate user information. However, with any investigation, the investigator's diligence is required to ensure the information is treated with both respect and caution.

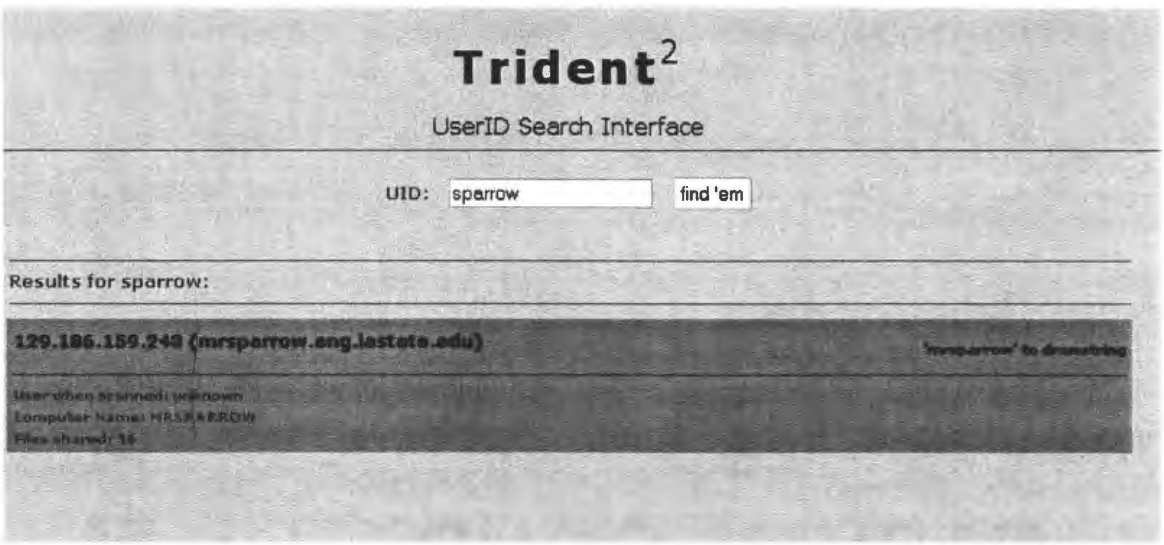

Figure 15 - User identity failure

Chapter 10. Future Work

Though Trident can access many different types of file sharing networks, there still are a few, such as Bit torrent and HTTP, which it cannot access. Work should be done to expand Trident's access to these networks. Second, the Gnutella module should be indexing file hash values when available. This would allow investigators to search by hash, as with Peer Precision, to quickly identify known child pornography.

Trident's database structure should also be revisited. Though the current structure, using MySQL's Full Text indexing system, is fairly efficient, it limits Trident to one database. Changing the way files are indexed, such as creating word indexes mapped to file index entries (Figure 16) would improve the search times over Trident's database tables. A word mapping table splits each file name into individual words, and then creates tables for each word. This allows the search engine to only query tables that match the search parameters, essentially searching a subset of the entire file index database. Individual file information can then be looked up via a unique ID assigned to each file. Due to Trident's immediate need, it was not initially created with this style of lookup in mind.

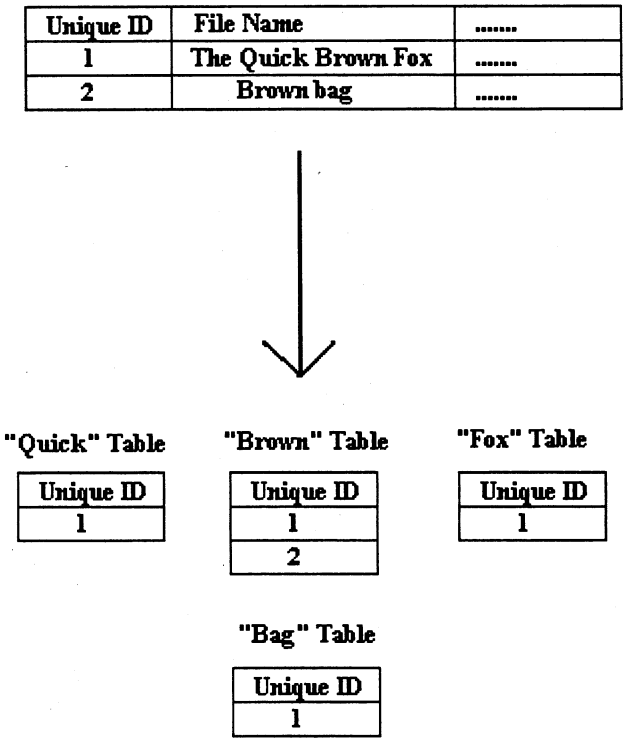

Figure 16 - Example word mapping table

In addition, an application should be developed to allow users to search through contents of text-based files. This would give investigators the ability to look for specific words within files available on the network. For instance, if a criminal has the words "BOMB THREAT" within a text file on the network, unless the file was named in a way investigators could locate it (such as "a bomb threat.txt"), there is no way for them to search for those specific terms. Such an application would have to download each file to create an index of it – which would take a lot of space and processing power. However, such an application could have many uses, from identifying the user of a computer to locating specific information about a crime.

Finally, a download manager should be developed, that can be configured to continually attempt to download material from computers. This would allow investigators to request a download from a computer not currently on the network. The download manager would keep trying to connect to the computer until it becomes available, at which time it would download the suspected illegal material for analysis. The download manager would then alert the investigator that the requested material has been obtained, at which time the investigator could analyze the material.

By implementing these ideas, Trident could become a better investigation tool. Such work should be considered immediately, to ensure investigators have every capability provided within Trident at their disposal.

Chapter 11. Conclusions

Computer crime is prevalent across many computer networks. Due to the newness of this crime, investigators lack sufficient tools to combat these types of offenses efficiently. Though the investigators have developed proactive solutions, the resource requirements for these solutions are often beyond what a typical department can afford. Therefore, an autonomous approach to these proactive solutions was developed, to combat the resource requirements, while giving investigators a new tool for investigations. This approach should satisfy four criteria – it should run without any intervention from investigators, it should preserve the integrity of the information it obtains, it should be easy to use, and it should be able to easily adapt to any type of network protocol.

One project, Trident, satisfies all four of these criteria, by being modular and web based. Trident is an indexing engine, a search engine and a browsing service wrapped up into one application. Its modularity allows Trident to easily expand into new network protocols, and since it is web based, users do not need to learn how to navigate through new applications. Trident's indexer continually crawls through computer networks, cataloging all material present on the network at all times of the day. The search engine allows investigators to quickly locate material through a common search environment. Its web based browsing service allows investigators to download and verify suspect material with the click of a mouse, giving greater accuracy to their investigations and a better basis for search warrants.

However, it should be recognized that a project like Trident has a limited utility life span. Once criminals become aware of Trident, and more importantly, how it works, the cat-and-mouse game begins between developments to cloak Trident's activities and attempts to block Trident's scanning. Investigators using Trident should be fully aware of this degradation in scanning, and never rely solely on tools such at Trident to conduct investigations.

Though computer crime is becoming common among today's Internet, projects such as Trident will help investigators track down and prosecute on-line criminals. Its presence is both a tool for investigators and a deterring presence for future criminals. In the future, perhaps these tools will help put an end to computer crime.

References

[1] Operation Peer Precision. Wyoming ICAC Task Force, October 2004.

[2] Internet Sex Crimes Against Minors: The Response of Law Enforcement. NCMEC, November 2003.

[3] "D03272": Combating Child Pornography. General Accounting Office. December, 2002.

[4] "Best Practices for Seizing Electronic Evidence v.2.0" (handbook). International Association of Chiefs of Police, 2002.

[5] "January 2005 Internet Usage". Nielsen//NetRatings. http://www.nielsen-netratings.com/reports.jsp?section=pub_reports&report=usage&period=monthly&panel_type=3. January 2005.